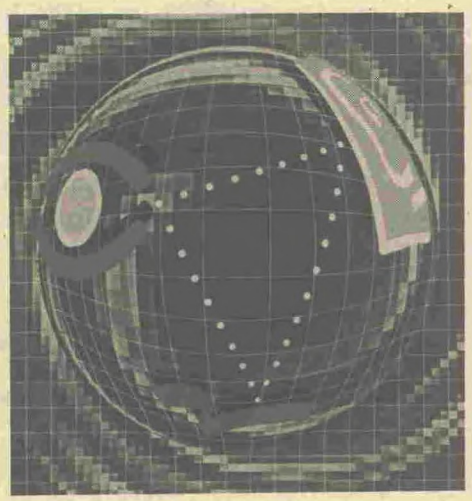
Unix

ไพโรจน์ ไววานิชกิจ

ก้าวแรก... สู่โลก UNIX ตอนที่ 2

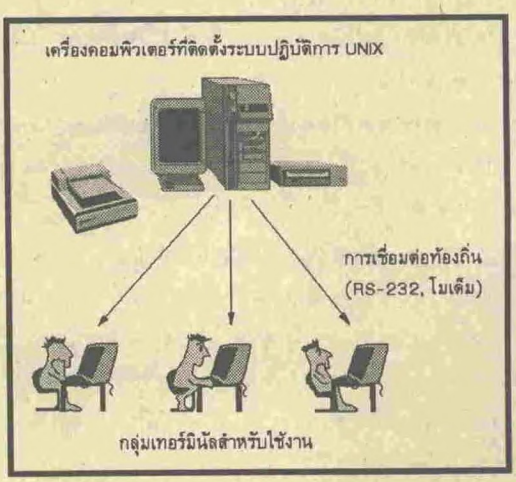


การติดต่อสื่อสารกับผู้ใช้งานคนอื่นที่ใช้งานอยู่บนเครื่องคอมพิวเตอร์หลักโดยใช้ชุดคำสั่งที่มีอยู่บนระบบปฏิบัติการ UNIX

หลังจากที่ผู้อ่านสามารถ login เข้าสู่ระบบปฏิบัติการ UNIX ได้แล้ว สิ่งต่อไปที่สมควรจะได้กล่าวถึงก็คือการติดต่อสื่อสารกับผู้ใช้งานคนอื่นซึ่งใช้งานอยู่บนเครื่องคอมพิวเตอร์หลักเดียวกัน โดยใช้ชุดคำสั่งที่มีอยู่บนระบบปฏิบัติการ UNIX และนี่คือสิ่งที่จะกล่าวถึงในบทความตอนนี้

ในโลกของความเป็นจริงมักจะพบว่ามีการติดต่อสื่อสารระหว่างผู้ใช้งานซึ่งอยู่ต่างเครื่องคอมพิวเตอร์กัน โดยอาศัยการเชื่อมต่อผ่านระบบเน็ตเวอร์ก ผู้เขียนจะกล่าวถึงการติดต่อสื่อสารข้ามเน็ตเวอร์กนี้แยกต่างหากในบทความตอนต่อ ๆ ไป สำหรับบทความตอนนี้ขอให้เข้าใจร่วมกันว่าผู้เขียนอธิบายเฉพาะการติดต่อสื่อสารระหว่างผู้ใช้งานตามโครงสร้างของระบบซึ่งแสดงในรูปที่ 1 เท่านั้น

โครงสร้างของการจัดใช้งานระบบปฏิบัติการ UNIX ดังรูปที่ 1 พบได้โดยทั่วไปกับองค์กรหรือหน่วยงานขนาดเล็ก ๆ ซึ่งใช้งานโปรแกรมประยุกต์ที่สามารถทำงานได้บนเครื่องคอมพิวเตอร์เพียงเครื่องเดียว และมีผู้ต้องการใช้งานโปรแกรมดังกล่าวพร้อม ๆ กันหลายคน การใช้จอภาพและคีย์บอร์ดของเครื่องคอมพิวเตอร์หลักซึ่งมีอยู่เพียงชุดเดียวจะไม่สามารถรองรับความต้องการในการใช้งานของแต่ละคนได้ทันท่วงที จึงจำเป็นต้องมีการติดต่ออุปกรณ์เทอร์มินัลสำหรับให้ผู้ใช้แต่ละคนสามารถติดต่อกับเครื่องคอมพิวเตอร์หลักได้ เทอร์มินัลโดยทั่วไปประกอบด้วย จอภาพ คีย์บอร์ด และส่วนของการติดต่อสื่อสาร ซึ่งมักจะเป็นพอร์ตเชื่อมต่อแบบ RS-232 ผู้ใช้งานสามารถเชื่อมต่อเทอร์มินัลของตนเข้ากับเครื่องคอมพิวเตอร์หลักได้โดยตรงโดยผ่านทางสายเชื่อมต่อแบบ RS-232 ในกรณีที่เทอร์มินัลของตนอยู่ใกล้ ๆ กับคอมพิวเตอร์หลักหรือต่อผ่านโมเด็มในกรณีที่ทำงานอยู่ต่างสถานที่กัน



รูปที่ 1 การติดต่อสื่อสารระหว่างผู้ใช้งานระบบ UNIX ซึ่งใช้งานเครื่องคอมพิวเตอร์หลักเดียวกัน

มองหาเพื่อนร่วมระบบ

เมื่อสามารถติดต่อกับระบบปฏิบัติการ UNIX ได้แล้ว ก่อนที่จะทำการติดต่อสื่อสารกับผู้ใช้งานรายอื่นก็คงจะต้องทราบเสียก่อนว่าในขณะนั้นมีใครติดต่ออยู่กับเครื่องคอมพิวเตอร์หลักบ้าง ระบบ UNIX มีชุดคำสั่งสำหรับตรวจสอบผู้ใช้งานอยู่ 3 คำสั่ง คือ **finger, w** และ **who** คำสั่ง **finger** ใช้สำหรับแสดงรายชื่อของผู้ใช้งานทั้งหมดที่เชื่อมต่ออยู่กับเครื่องคอมพิวเตอร์หลัก ก่อนจะได้อธิบายต่อไป ขอให้ผู้อ่านดูผลที่ได้จากการพิมพ์คำสั่ง finger ผ่านระบบ UNIX

```
$ finger
Login        Name              TTY  Idle   When      Where
pairoj   Pairoj Waiwanijchakij *p0    1    Sat 7:01  mammoth01ba
thisun   Thisun Polasup         02  23:21  Fri 22:39 mammoth01ba
kamol    Kamol Kaemarungsi      03         Sat 6:45  mammoth01ba

$
```

เมื่อป้อนคำสั่ง **finger** ระบบ UNIX จะแสดงรายชื่อของผู้ใช้งานที่ใช้งานระบบ UNIX อยู่ในขณะนั้น โดยรายงานข้อมูลต่าง ๆ เป็นคอลัมน์ดังนี้

- Login แสดง login account ของผู้ใช้งานแต่ละคน
- Name แสดงชื่อและนามสกุลจริงของผู้ใช้งานแต่ละคน
- TTY (Teletype port) เนื่องจากระบบ UNIX สามารถให้บริการผู้ใช้งานได้หลาย ๆ คนพร้อมกัน การอ้างถึงผู้ใช้งานแต่ละคนจะอยู่ในรูปของเลขหมาย TTY ซึ่งอาจจะมองง่าย ๆ ว่าเป็นคล้ายกับเลขที่ของพอร์ตสื่อสารที่ผู้ใช้รายนั้นเชื่อมต่ออยู่ เช่น ผู้ใช้งานชื่อ thisun เชื่อมต่อเทอร์มินัลของตนเข้ากับพอร์ตสื่อสารหมายเลข 2 จึงถูกอ้างถึงเป็น TTY 02 อย่างไรก็ตาม ความหมายของ TTY ตามความเป็นจริงมีความลึกซึ้งกว่านี้ ผู้เขียนจะได้กล่าวถึงอีกครั้งเมื่อพูดถึงการทำงานกับเชลล์

ถ้าสังเกตที่ pairoj จะเห็นว่ามีเครื่องหมายดอกจัน (*) ปรากฏอยู่หน้าเลข TTY เครื่องหมายดังกล่าวแสดงให้ทราบว่าผู้ใช้งานปิดการรับข่าวสารจากผู้อื่นโดยใช้คำสั่ง **mesg** อันจะได้กล่าวถึงต่อไป

- Idle แสดงระยะเวลานับจากการป้อนคำสั่งให้ระบบ UNIX ครั้งล่าสุดจนถึงปัจจุบันของผู้ใช้รายนั้น
- When แจ้งว่าผู้ใช้ login เข้าสู่ระบบตั้งแต่เมื่อไร
- Where แจ้งว่าผู้ใช้งาน login เข้าสู่ระบบ UNIX โดยมาจากเครื่องคอมพิวเตอร์ใด ในกรณีนี้มีเครื่องคอมพิวเตอร์หลักเพียงเครื่องเดียว ผู้ใช้งานทุกคนจึงมีค่า Where เดียวกัน

หากต้องการทราบรายละเอียดของผู้ใช้รายใดรายหนึ่งมากกว่าที่ปรากฏบนจอภาพตามที่ได้กล่าวมาแล้ว ผู้ใช้งานสามารถป้อนคำสั่ง **finger** ตามด้วย login account ของผู้ใช้งานที่ต้องการทราบ โดยจะได้รับผลแจ้งกลับจากระบบ UNIX ดังนี้

```
$ finger thisun
Login name: thisun                      In real life: Thisun Polasup
Directory: /home/usergroup1/thisun      Shell: /bin/csh
On since Aug 15 22:39:20 on tty02
New mail received Fri Aug 15 20:15:02 1997;
   unread since Wed Aug 13  12:03:56 1997
Plan:
Will leave to Finland on next week.

$
```

สังเกตตั้งแต่บรรทัด Plan: ซึ่งเป็นการแจ้งหมายเหตุต่าง ๆ ที่ thisun ต้องการแจ้งให้ผู้อื่นทราบ โดยในกรณีที่ผู้ใช้งานต้องการแทรกข่าวสารเหล่านี้ให้กับผู้อื่นก็สามารถทำได้โดยการสร้างไฟล์ชื่อ **.plan** ในไดเรคทอรีใช้งานของตน ซึ่งในกรณีของ thisun ก็คือการสร้างไฟล์ในไดเรคทอรี /home/usergroup1/thisun (หมายเหตุ : ตำแหน่งไดเรคทอรีที่ผู้เขียนใช้เป็นตัวอย่างเป็นการสมมติขึ้น ในความเป็นจริงอาจเป็นอย่างอื่นก็ได้ ขึ้นอยู่กับผู้ดูแลระบบ UNIX นั้น ๆ จะเป็นผู้ออกแบบ)

สำหรับคำสั่ง w จะใช้ในการแสดงรายชื่อของผู้ใช้งานระบบ UNIX โดยแสดงในรูปแบบของพฤติกรรมการใช้งานระบบดังนี้

```
$ w
  7:02am up 15 days, 10.22, 3 users,load average: 0.15, 0.03, 0.02
User   tty      login@          idle     JCPU     PCPU     what
pairoj ttyp0    7:01am          1        11         1      w
thisun tty02    Fri 10pm        23:21     16        1      -sh
kamol  tty03    6:45am                                     vi .profile

$
```

ความแตกต่างระหว่างข้อมูลที่แสดงจากคำสั่ง **finger** กับคำสั่ง w ก็คือ คำสั่ง w มีการแสดงค่าเวลาที่ซีพียูของคอมพิวเตอร์หลักทำงานให้กับผู้ใช้งานแต่ละราย นอกจากนั้นยังมีการระบุชื่อชุดคำสั่งซึ่งผู้ใช้แต่ละรายกำลังใช้งานอยู่ สำหรับข้อความเฉพาะที่ปรากฏในบรรทัดบนสุดจะเป็นสิ่งเดียวกับผลลัพธ์ที่ได้จากการป้อนคำสั่ง **uptime** โดยเป็นการแสดงถึงเวลาปัจจุบันในขณะนั้น ระยะเวลาที่เครื่องคอมพิวเตอร์หลักได้ทำงานมาตั้งแต่เริ่มเปิดเครื่องครั้งล่าสุด โดยแสดงเป็นจำนวนวันและเวลา (จากตัวอย่างแสดงว่าเครื่องคอมพิวเตอร์หลักได้ทำงานมาเป็นเวลา 15 วัน กับอีก 10 ชั่วโมง 22 นาที) นอกจากนี้ยังมีการแสดงจำนวนผู้ที่ใช้งานเครื่องคอมพิวเตอร์หลักอยู่ สำหรับตัวเลขสามชุดสุดท้ายแสดงถึงค่าเฉลี่ยของจำนวนงานที่รอเข้าคิวป้อนให้ซีพียูในปัจจุบัน ก่อนหน้านี้ 5 นาที และ 15 นาที ตามลำดับ

สำหรับคำสั่ง **who** จะให้ผลลัพธ์คล้ายกับคำสั่ง w เพียงแต่มีปริมาณข้อมูลน้อยกว่าเท่านั้น ในกรณีที่ผู้ใช้งานระบบ UNIX ต้องการทราบว่าตนกำลังใช้งานเทอร์มินัลหมายเลขใดและตนได้ทำการ login เข้าสู่ระบบตั้งแต่เมื่อใด สามารถตรวจสอบได้โดยใช้คำสั่ง **who am i** ดังนี้

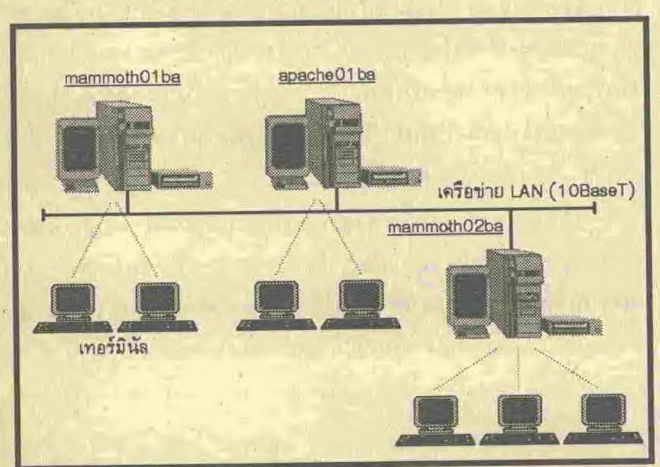
```
$ who am i
pairoj  ttyp0          Aug 16  7:01

$
```

เริ่มพูดคุย

ในกรณีที่ผู้ใช้งานต้องการทำการสนทนากับผู้ใช้งานรายอื่นที่ใช้งานอยู่บนเทอร์มินัลอื่น ๆ หรือแม้ในกรณีที่เป็นการติดต่อข้ามผ่านเครือข่าย ระบบปฏิบัติการ UNIX มีชุดคำสั่งที่สนับสนุนการสื่อสารระหว่างผู้ใช้งานอยู่หลายคำสั่งด้วยกัน ในที่นี้จะได้กล่าวถึง talk, write, mesg และ mail ก่อนอื่นผู้เขียนขอตกลงกับผู้อ่านก่อนว่าหลังจากนี้ไปการสมมติตัวอย่างประกอบการใช้งานชุดคำสั่งต่าง ๆ ที่เกี่ยวข้องกับการสื่อสารระหว่างผู้ใช้งานระบบปฏิบัติการ UNIX จะเป็นในรูปของการสื่อสารภายในระบบเครือข่ายคอมพิวเตอร์

ระบบเครือข่ายคอมพิวเตอร์คือการเชื่อมต่อเครื่องคอมพิวเตอร์เข้าด้วยกันเพื่อประโยชน์ในด้านของการแลกเปลี่ยนข้อมูลระหว่างกัน ตัวอย่างหนึ่งที่เห็นได้ชัดก็คือการเชื่อมต่อเครื่องคอมพิวเตอร์หลาย ๆ เครื่องเข้าด้วยกันเป็นเครือข่ายโดยใช้เครือข่าย LAN ผู้เขียนขอใช้รูปที่ 2 เป็นหลักในการอ้างประกอบตัวอย่างต่าง ๆ ในที่นี้จะเห็นว่ามีเครื่องคอมพิวเตอร์หลัก 3 เครื่องเชื่อมต่อถึงกันโดยเครือข่าย LAN แต่ละเครื่องต่างก็ยังคงมีการเชื่อมต่อกับเทอร์มินัลจำนวนหนึ่ง ในทางปฏิบัติจะพบว่าผู้ใช้งานซึ่ง login เข้าสู่เครื่องคอมพิวเตอร์หลักเครื่องหนึ่งจะสามารถเข้าถึงระบบไฟล์ของเครื่องคอมพิวเตอร์หลักเครื่องอื่น ๆ ซึ่งต่ออยู่ภายในระบบเครือข่ายเดียวกันได้ ผู้เขียนจะขอกล่าวถึงเรื่องนี้โดยละเอียดในภายหลัง

คำสั่ง **talk** ใช้สำหรับการติดต่อสื่อสารระหว่างผู้ใช้งาน 2 รายในลักษณะที่สามารถโต้ตอบกันได้โดยตรงในเวลาเดียวกัน (Interactive) โดยการสื่อสารสามารถเกิดขึ้นระหว่างผู้ใช้งาน 2 รายที่ติดต่อกับเครื่องคอมพิวเตอร์หลักเครื่องเดียวกันหรือต่างเครื่องกันก็ได้ ในกรณีที่ต้องการติดต่อกับผู้ใช้งานที่อยู่บนเครื่องคอมพิวเตอร์หลักเครื่องเดียวกันสามารถกระทำได้โดยใช้คำสั่ง **talk** [login account] เช่น **talk thisun** เป็นต้น แต่หากต้องการติดต่อกับผู้ใช้งานซึ่งใช้งานอยู่บนเครื่องคอมพิวเตอร์หลักเครื่องอื่น จำเป็นจะต้องระบุชื่อของเครื่องคอมพิวเตอร์หลักปลายทางลงไปด้วย เช่น **talk suthida@apache01ba**



รูปที่ 2 เครือข่ายคอมพิวเตอร์

เมื่อป้อนคำสั่งไปแล้ว ผู้ป้อนคำสั่งจะเห็นว่าหน้าจอของตนจะถูกแบ่งออกเป็น 2 ส่วน โดยหลังจากผู้ใช้งานที่ต้องการติดต่อด้วยตอบรับการขอสนทนาแล้ว แต่ละฝ่ายจะเห็นว่าหน้าจอส่วนบนคือข่าวสารที่ตนป้อนผ่านคีย์บอร์ดเพื่อส่งไปยังคู่สนทนาที่อยู่อีกเครื่องหนึ่ง สำหรับส่วนล่างของจอภาพก็คือข่าวสารที่ตนได้รับการตอบกลับมา ขั้นตอนในการสร้างการสื่อสารเริ่มจากผู้ใช้งานรายหนึ่งต้องการสนทนากับผู้ใช้งานอีกรายหนึ่ง ในกรณีนี้จะสมมติให้ทั้ง 2 ฝ่ายใช้งานอยู่บนคอมพิวเตอร์หลักต่างเครื่องกัน โดยคมสัณห์ซึ่งใช้งานอยู่บนเครื่อง mammoth01ba ต้องการติดต่อกับสุทธิดาซึ่งใช้งานเครื่อง apache01ba

```
$ talk suthida@apache01ba
```

ทันทีที่คมสัณห์ป้อนคำสั่งนี้และกดปุ่ม [ENTER] บนหน้าจอเทอร์มินัลของคมสัณห์จะถูกแบ่งออกเป็น 2 ส่วนและจะปรากฏข้อความ [Waiting for your party to respond] บนหน้าจอแจ้งว่ากำลังรอการตอบรับจากผู้ใช้ปลายทาง ในขณะเดียวกับที่บนหน้าจอเทอร์มินัลของสุทธิดาจะปรากฏข้อความแจ้งว่ามีการขอติดต่อจากคมสัณห์ดังนี้

```
Messsage from Talk_Daemon@mammoth01ba at 7:12 ...
talk : connection requested by komsarn@mammoth01ba.
talk : respond with: talk komsarn@mammoth01ba
$ talk komsarn@mammoth01ba
```

เมื่อสุทธิดาตัดสินใจตอบรับการติดต่อจากคมสัณห์ ก็จะพิมพ์คำสั่ง talk เช่นเดียวกัน ซึ่งหลังจากป้อนคำสั่งไปแล้ว บนหน้าจอของสุทธิดาก็จะถูกแบ่งออกเป็นส่วนเช่นเดียว นับจากนี้ไปผู้ใช้งานทั้งสองรายก็สามารถที่จะพิมพ์ข้อความเพื่อติดต่อสื่อสารกันได้โดยตรง ทั้งนี้ตัวอย่างข้อความการสนทนาที่ปรากฏบนหน้าจอของผู้ใช้แต่ละรายจะมีลักษณะเป็นดังนี้

หน้าจอของคมสัณห์

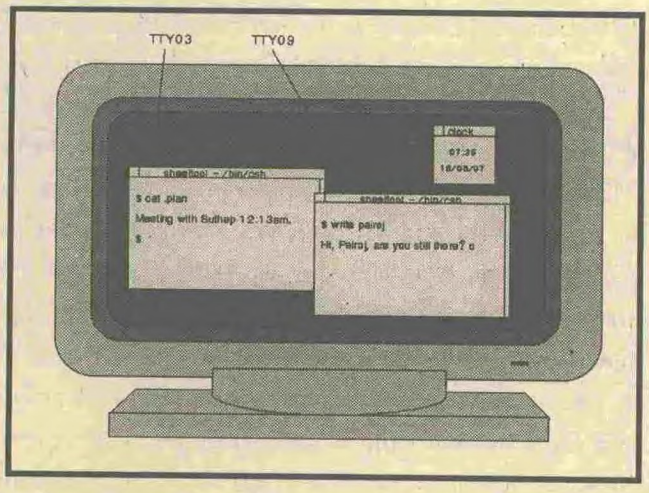
```
Did you finish the slides for the 9:30
 meeting today?
Sounds goog, see you in a few minutes!


- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Hi, komsarn, what's up?
Yes, they're all set.  Sould I just meet you
in the conference room?
Bye.
```

หน้าจอของสุทธิดา

```
[Connection established]
Hi, komsarn, what's up?
Yes, they're all set.  Sould I just meet you
in the conference room?
Bye.

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - -
Did you finish the slides for the 9:30
 meeting today?
Sounds goog, see you in a few minutes!
```

เมื่อฝ่ายใดฝ่ายหนึ่งต้องการที่จะยุติการสนทนาก็สามารถทำได้โดยการกดปุ่ม [DELETE] โปรแกรม talk ทางด้านของตนก็จะหยุดทำงานและกลับเข้าสู่เชลล์อีกครั้งหนึ่ง ทางด้านของคู่สนทนาจะได้รับแจ้งว่าผู้ใช้งานที่อยู่ปลายทางอีกด้านหนึ่งก็จะได้รับแจ้งว่าการสนทนาสิ้นสุดลงเช่นเดียวกัน สำหรับในบางเวลา ผู้ใช้งานบางคนกำลังจดจ่ออยู่กับการทำงานของตนและไม่ต้องการถูกรบกวนจากการขอสนทนาจากผู้อื่น ระบบปฏิบัติการ UNIX มีคำสั่งซึ่งผู้ใช้งานสามารถกำหนดให้ระบบป้องกันตนจากการขอสนทนาได้ คำสั่งดังกล่าวคือ **mesg** เมื่อป้อนคำสั่งนี้แล้วผู้ใช้งานจะไม่ได้รับข้อความที่ถูกส่งมาจากผู้อื่นได้ การป้อนคำสั่ง **mesg n** จะทำให้ระบบ UNIX ป้องกันการรับข่าวสารจากผู้อื่น ในกรณีที่ต้องการรับข่าวสารตามเดิมก็สามารถทำได้โดยการป้อนคำสั่ง **mesg y** คำสั่ง **mesg** ใช้ป้องกันการรับข่าวสารที่เกิดขึ้นจากการที่ผู้ใช้รายอื่นใช้คำสั่ง **talk** ดังได้กล่าวมาแล้ว และคำสั่ง **write** อันจะได้กล่าวถึงเป็นอันดับต่อไป

คำสั่ง **write** นับเป็นชุดคำสั่งของระบบปฏิบัติการ UNIX ที่เกิดขึ้นก่อนหน้าคำสั่ง **talk** มาเป็นเวลานาน รูปแบบการใช้คำสั่ง **write** ที่ง่ายที่สุดก็คือการป้อนคำสั่งดังกล่าวตามด้วย login account ของผู้ใช้งานที่ต้องการจะติดต่อด้วย ในกรณีที่ผู้ใช้งานที่ต้องการติดต่อด้วยนั้นเปิดใช้ TTY มากกว่า 1 ชุด อาจจะต้องมีการระบุด้วยว่าต้องการติดต่อกับ TTY หมายเลขใดเพิ่มลงไปในคำสั่งด้วย ผู้เขียนขอขยายความเพิ่มเติมว่าผู้ใช้งานแต่ละรายสามารถที่จะเปิดใช้งานเชลล์ได้มากกว่าหนึ่งชุด ตามที่ได้กล่าวมาในช่วงต้นว่า TTY เลขหนึ่งอาจมองเป็นเทอร์มินัลเครื่องหนึ่ง แต่ถ้าจะกล่าวถึงให้ชัดลงไปก็คือ TTY เลขหนึ่งจะใช้แทนเชลล์หนึ่งเชลล์ของผู้ใช้งานแต่ละราย ในกรณีของระบบปฏิบัติการ UNIX รุ่นที่เป็นเท็กซ์โหมด ผู้อ่านจะมองเห็นภาพไม่ชัด ขอให้ลองนึกถึงระบบปฏิบัติการ UNIX รุ่นที่เป็นแบบกราฟิก (GUI) ซึ่งมีรูปแบบการแสดงผลคล้ายคลึงกับระบบปฏิบัติการ Windows 95 ซึ่งผู้ใช้งานสามารถเปิดวินโดว์ที่เป็นเชลล์ได้มากเท่าใดก็ได้ดังรูปที่ 3



รูปที่ 3 การเปิดเชลล์วินโดว์มากกว่า 1 เชลล์บนระบบปฏิบัติการ UNIX แบบกราฟิก

รูปแบบของคำสั่ง **write** คือ **write *login account ของผู้ใช้งานที่ต้องการติดต่อ [เลขหมาย TTY]***

กลับมาที่ตัวอย่างของการติดต่อสื่อสารระหว่างผู้ใช้งานซึ่งใช้งานเทอร์มินัลแบบเท็กซ์โหมดตามเดิม สมมติว่าผู้ใช้งานที่ชื่อคมสัณห์ต้องการติดต่อกับผู้เขียนโดยใช้คำสั่ง **write** หากคมสัณห์เพียงแต่ต้องการส่งข่าวสารบางอย่างโดยไม่สนใจว่าผู้เขียนจะว่างอยู่หรือไม่ หรือไม่ตั้งใจว่าจะสนทนากับผู้เขียนเป็นกิจลักษณะ ก็สามารถที่จะป้อนคำสั่ง write ระบุ login account ของผู้เขียน แล้วพิมพ์ข้อความที่ต้องการจะส่งจนครบ แล้วจึงกดปุ่ม Control พร้อมกับปุ่ม D เป็นการส่งข่าวสารทั้งหมดพร้อมกับกลับคืนสู่เชลล์ตามเดิม ทางด้านของผู้เขียนเมื่อได้รับข่าวสารดังกล่าวปรากฏขึ้นบนหน้าจอ หากต้องการตอบกลับก็จะทำในสิ่งเดียวกันกับคมสัณห์ จะเห็นได้ว่าหากจำเป็นจะต้องมีการสนทนากันระหว่างทั้งสองด้วยวิธีการดังกล่าวก็คงเป็นสิ่งที่ยุ่งยากวุ่นวายมาก เพราะแต่ละฝ่ายจะตอบกลับแต่ละครั้งก็ต้องพิมพ์คำสั่ง **write** กันใหม่

อันที่จริงแล้วเราสามารถเพิ่มประสิทธิภาพของการใช้คำสั่ง **write** ในกรณีของการสนทนาแบบออนไลน์ได้ ทั้งนี้เนื่องจากคำสั่ง **write** จะส่งข้อความไปยังปลายทางได้ทีละบรรทัดหลังจากผู้พิมพ์กดปุ่ม ENTER ซึ่งคู่สนทนายังไม่จำเป็นต้องกดปุ่ม Control และปุ่ม D หากคิดว่าต้องการจะสนทนาแบบออนไลน์กับผู้ใช้งานอีกคนหนึ่ง ตัวอย่างต่อไปนี้คือการสนทนาระหว่างคมสัณห์และผู้เขียน โดยแต่ละฝ่ายป้อนคำสั่ง **write** เพียงครั้งเดียว เริ่มจากการที่คมสัณห์ป้อนคำสั่ง **write** ระบุว่าต้องการส่งข่าวสารมายังผู้เขียนพร้อมกับป้อนข่าวสารชุดแรก

```
$ write pairoj
Hi, Pairoj, are you still there? o
```

สังเกตว่ามีการจบข้อความชุดแรกด้วยตัวอักษร "O" ซึ่งอันที่จริงก็มิใช่คำสั่งหรือพารามิเตอร์พิเศษแต่ประการใด การระบุตัวอักษรไม่ว่าจะเป็นอักษรอะไรก็ตามปิดท้ายข้อความเป็นเพียงการแสดงให้คู่สนทนาปลายทางทราบว่าผู้ส่งข่าวสารรอให้ฝ่ายตรงข้ามตอบกลับมา เมื่อผู้เขียนได้รับข่าวสารจากคมสัณห์แล้วและต้องการที่จะร่วมสนทนาด้วย ก็จะป้อนคำสั่ง **write** ระบุ login account ของคมสัณห์พร้อมทั้งป้อนข้อความที่ต้องการจะตอบกลับ ทางด้านหน้าจอเทอร์มินัลของคมสัณห์จะปรากฏข้อความดังนี้ (แสดงข้อความที่ได้รับด้วยอักษรตัวเอน)

```
$ write pairoj
Hi, Pairoj, are you still there? o
   Message from Pairoj on mammoth01ba (ttyp0)  (Sat Aug 16
07:25]...
Yes Komsarn, I'm here and OK. o
```

เพื่อป้องกันความสับสนของผู้ใช้งาน ระบบปฏิบัติการ UNIX จะแจ้งการได้รับข่าวสารจากคู่สนทนาพร้อมทั้งระบุวันเวลาที่ได้รับในบรรทัดก่อนหน้าข้อความที่ได้รับทุกครั้ง ในกรณีที่การสนทนาได้ดำเนินไปจนต่างฝ่ายต่างสิ้นสุดข่าวสารที่จะส่งให้แก่กันแล้ว ผู้ใช้งานฝ่ายใดฝ่ายหนึ่งจะเป็นฝ่ายขอออกจากคำสั่ง **write** ซึ่งก็ยังคงทำโดยการกดปุ่ม Control และปุ่ม D พร้อม ๆ กัน สำหรับตัวอักษร "OO" ซึ่งปิดท้ายข้อความสุดท้ายก็เป็นเพียงการแจ้งสิ้นสุดข่าวสารระหว่างคู่สนทนาเท่านั้น ระบบ UNIX มิได้รับทราบหรือสนใจแต่ประการใด

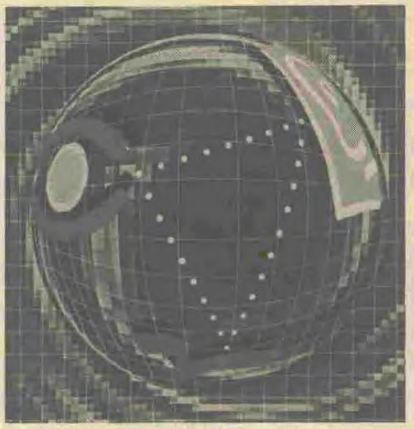
```
$ write pairoj
Hi, Pairoj, are you still there? o
   Message from Pairoj on mammoth01ba (ttyp0)  (Sat Aug 16 07:25]...
Yes Komsarn, I'm here and OK. o
.
.
Thank you, Pairoj - Bye oo

$ Bye, Komsarn oo
<EOT>


$
```

อิเล็กทรอนิกส์เมล์กับ UNIX

อิเล็กทรอนิกส์เมล์หรือจดหมาย (e-mail) เป็นยูทิลิตี้ที่ได้รับความนิยมใช้งานอย่างมากในวงการ UNIX และในเวลาต่อมาก็ได้เป็นหัวใจสำคัญของการใช้งานเครือข่ายอินเทอร์เน็ต ด้วยความสามารถของระบบปฏิบัติการ UNIX ซึ่งสามารถทำตัวเป็นที่ทำการไปรษณีย์สำหรับรับฝากและบริการส่งข่าวสารระหว่างผู้ใช้งานทั้งที่อยู่ภายในเครื่องคอมพิวเตอร์หลักเครื่องเดียวกันหรืออยู่ต่างเครือข่ายต่างพื้นที่กัน ในปัจจุบันมาตรฐานการสื่อสารโดยใช้อิเล็กทรอนิกส์เมล์ได้รับการขยายวงกว้างให้สามารถใช้งานได้ทั้งบนระบบปฏิบัติการ UNIX ทุก ๆ ตระกูล ไปจนถึงระบบปฏิบัติการชนิดอื่น ๆ เช่น DOS, Windows 95 หรือ Windows NT

ผู้ใช้งานระบบปฏิบัติการ UNIX สามารถใช้ส่งหรือรับจดหมายของตนได้โดยการป้อนคำสั่ง **mail** (ในบางระบบอาจให้ป้อนเป็น **Mail**) คำสั่ง **mail** จะแจ้งให้ UNIX เรียกโปรแกรมสนับสนุนการรับส่งจดหมายเพื่อให้บริการต่อผู้ใช้งาน การป้อนคำสั่ง **mail** ตามด้วย login account ของผู้ใช้งานรายอื่น ๆ จะหมายถึงการส่งจดหมายไปยังผู้ใช้งานรายนั้น ซึ่งอันที่จริงแล้ว login account ของผู้ใช้งานจะถูกเขียนให้อยู่ในรูปของจดหมายแอดเดรสในกรณีที่ต้องการส่งไปยังผู้รับซึ่งใช้งานอยู่ต่างเครื่องคอมพิวเตอร์กัน (อาจอยู่ในเครือข่ายเดียวกันหรือต่างเครือข่ายกันก็ได้) รูปแบบการป้อนคำสั่ง **mail** เพื่อการส่งจดหมายจะเป็นดังนี้

```
$ mail thisun
$ mail suthida@apache01ba
$ mail Micro@Se-Ed.com
```



ตัวอย่างต่อไปนี้คือการส่งอิเล็กทรอนิกส์เมล์โดยผู้เขียนไปยังตัวผู้เขียนเอง จะเห็นว่าเมื่อป้อนคำสั่ง mail pairoj และกดปุ่ม ENTER แล้ว UNIX จะขึ้นหัวข้อ Subject เพื่อให้ผู้ใช้งานป้อนหัวเรื่องของจดหมาย หลังจากป้อนหัวเรื่องแล้วผู้ใช้งานก็จะสามารถพิมพ์ข้อความที่ตนต้องการจะส่งลงไปจนกว่าจะครบใจความทั้งหมด ในกรณีตัวอย่างเป็นข้อความสั้น ๆ เพียงประโยคเดียว หลังจากจบข้อความแล้ว ผู้ใช้งานจะต้องกดปุ่ม Control และปุ่ม D พร้อม ๆ กันเพื่อแจ้งให้ระบบ UNIX ทราบว่าตนสิ้นสุดการป้อนข่าวสารแล้ว พร้อมทั้งต้องการให้ระบบ UNIX ส่งข้อความไปยังผู้รับปลายทาง และยุติการใช้งานโปรแกรม **mail**

```
$ mail pairoj
Subject: Test Message
This is a test message that I am sending myself.
[CONTROL-D]
$
```

สำหรับการอ่านจดหมายที่ตนได้รับจากผู้อื่นนั้น ผู้ใช้งานระบบ UNIX สามารถป้อนคำสั่ง **mail** โดยไม่มีชื่อใด ๆ ต่อท้ายโดยจะได้รับแจ้งข้อมูลจากระบบ UNIX ดังแสดงในตัวอย่างต่อไปนี้ จะเห็นว่าสองบรรทัดแรกที่ UNIX แสดงผลจะเป็นการระบุรุ่นของโปรแกรม mail ที่ใช้งานอยู่ พร้อมทั้งคำแนะนำสำหรับการขอความช่วยเหลือต่าง ๆ ในบรรทัดแรก สำหรับบรรทัดที่สองจะระบุตำแหน่งที่เก็บไฟล์ข่าวสารที่เป็นของผู้ใช้งาน ซึ่งจะเห็นว่าไฟล์อิเล็กทรอนิกส์เมล์ของผู้เขียนถูกเก็บไว้ในไฟล์ชื่อ **/usr/spool/mail/pairoj** ตามด้วยจำนวนจดหมายที่มีอยู่ ซึ่งแยกระบุว่ามีจดหมายกี่ฉบับที่ยังไม่ถูกเปิดอ่าน สำหรับส่วนที่เหลือจะเป็นการแสดงลำดับของจดหมายแต่ละฉบับที่ผู้ใช้งานได้รับ โดยจะเห็นว่าบางฉบับจะมีตัวอักษร N กำกับอยู่ ซึ่งหมายความว่าเป็นฉบับที่ยังไม่ถูกเปิดอ่าน นอกจากนี้สำหรับทุก ๆ ฉบับจะมีการระบุชื่อผู้ส่ง วันเวลาที่ส่ง และความยาวของจดหมาย (จำนวนบรรทัด / จำนวนตัวอักษร) พร้อมทั้งหัวเรื่องของจดหมาย จดหมายฉบับที่มีเครื่องหมาย ">" ปรากฏอยู่จะเป็นจดหมายฉบับที่ถูกอ้างถึงในปัจจุบัน สังเกตว่าเครื่องหมายพรอมต์ที่ปรากฏจะเป็นรูป & ซึ่งแสดงว่าผู้ใช้งานกำลังอยู่ในโปรแกรม mail มิได้อยู่ในเชลล์ UNIX

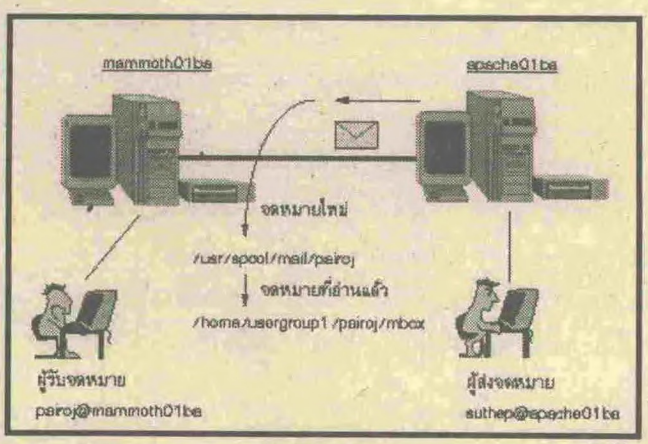
```
$ mail
Mail version SMI 4.1-OW3 Sat Aug 16 7:45:24 PDT 1997  Type ? for help.
"/usr/spool/mail/pairoj": 3 messages 2 new
>N  1   pairoj  Sat  Aug  16  07:39     6/120   Test Message
 N  2   thisun  Wed  Aug 13  17:10     6/115   Shall we go?
    3   suthep  Mon  Aug 11   08:1     20/1113 New Product
&
```

ในกรณีที่ต้องการอ่านจดหมายฉบับที่ถูกอ้างถึงอยู่ด้วย เครื่องหมาย ">" ก็สามารถทำได้โดยการกดปุ่ม ENTER ซึ่งจะทำให้ UNIX เปิดอ่านข้อความในจดหมายและแสดงผลปรากฏขึ้นบนหน้าจอ ดังนี้

```
& [ENTER]
Message 1:
From pairoj  Sat    Aug  16  07:39  PDT  1997
To: pairoj
Subject: Test Message
Status: R

This is a test message that I am sending myself.

&d [Enter]
```

จะเห็นว่า 4 บรรทัดแรกเป็นการแสดงรายละเอียดต่าง ๆ เกี่ยวกับผู้ส่ง หัวเรื่องของจดหมาย และสถานภาพ (R=Read, N=New เป็นต้น) อย่างไรก็ดี รายละเอียดต่าง ๆ เหล่านี้อาจจะไม่เหมือนกันสำหรับระบบ UNIX แต่ละระบบ ทั้งนี้ขึ้นอยู่กับการออกแบบของผู้ดูแลระบบเป็นหลัก เมื่อผู้ใช้งานอ่านจดหมายเรียบร้อยแล้วและต้องการจะลบออกจากฮาร์ดดิสก์ก็สามารถทำได้โดยพิมพ์ d พร้อมกับกดปุ่ม ENTER หากผู้ใช้อ่านจดหมายไปแล้วแต่ยังไม่ตัดสินใจลบออก จดหมายฉบับที่ถูกอ่านแล้วจะถูกส่งไปเก็บไว้ในไฟล์ชื่อ **mbox** ในไดเรคทอรีของผู้ใช้งาน

ในกรณีที่ผู้ใช้งานต้องการทราบชุดคำสั่งที่มีอยู่ในโปรแกรม mail ก็สามารถตรวจสอบได้โดยการพิมพ์เครื่องหมาย ? เมื่ออยู่ในพรอมต์ & โปรแกรม mail จะแสดงรายการชุดคำสั่งพร้อมทั้งคำอธิบายให้กับผู้ใช้



รูปที่ 4 การส่งและรับอิเล็กทรอนิกส์เมล์โดยผู้ใช้งานในระบบปฏิบัติการ UNIX

กลับมาที่โปรแกรม mail อีกครั้ง หลังจากที่ได้อ่านจดหมายจบลงแล้ว ผู้ใช้งานจะพบว่าตนเองได้กลับมาที่ตำแหน่งพรอมต์ & อีกครั้งหนึ่ง ในกรณีที่ต้องการอ่านจดหมายฉบับอื่นก็สามารถทำได้โดยการพิมพ์เลขหมายซึ่งกำกับอยู่หน้าจดหมายฉบับนั้น ในกรณีตัวอย่างของผู้เขียนซึ่งต้องการอ่านจดหมายที่ส่งมาจากไปรษณีย์ก็สามารถทำได้โดยการพิมพ์ 2 พร้อมกับกด ENTER ซึ่งจะเป็นการแจ้งให้โปรแกรม mail เปิดอ่านจดหมายฉบับที่ 2 เมื่อถึงจุดนี้อาจจะมีผู้อ่านทักท้วงว่าหากผู้อ่านได้ทำการลบจดหมายฉบับแรกไปแล้ว เลขอ้างอิงจดหมายฉบับต่อไปควรจะร่นขึ้นเป็นเลข 1 แทนที่จดหมายฉบับแรกที่ถูกลบไปแล้ว ผู้เขียนขอชี้แจงว่าไม่ว่าผู้ใช้งานจะทำการลบจดหมายไปกี่ฉบับก็ตาม แต่ตราบใดที่ตนยังไม่ออกจากโปรแกรม mail ลำดับของจดหมายจะยังคงเป็นอยู่อย่างเดิม ต่างกันเพียงว่าจดหมายฉบับที่ถูกลบจะถูกนำหน้าด้วยตัวอักษร D (Delete) ซึ่งผู้ใช้งานสามารถทำการกู้กลับคืนมาได้หากต้องการ รูปที่ 4 เป็นการแสดงกระบวนการอ่านจดหมายของผู้ใช้งานระบบ UNIX

ในกรณีตัวอย่าง สมมติว่าหลังจากผู้เขียนอ่านจดหมายฉบับที่สองจบลงแล้วและต้องการจะกลับออกจากโปรแกรม mail ไปยังเชลล์ดำเนินการกับจดหมายทั้งหมดดังนี้

```
Saved 1 message in /home/usergroup1/pairoj/mbox
```

หมายความว่าโปรแกรม mail จะไม่สนใจจดหมายฉบับแรกที่ผู้เขียนอ่านและลบทิ้งไปแล้ว แต่จะย้ายจดหมายฉบับที่สองไปเก็บไว้ในไฟล์ **/home/usergroup1/pairoj/mbox** แทน

ในกรณีที่ต้องการส่งจดหมายไปให้บุคคลมากกว่าสองคนขึ้นไปก็สามารถทำได้โดยการป้อนคำสั่ง **mail** ตามด้วย login account ของผู้ที่ต้องการจะส่งจดหมายไปถึง โดยเขียนเรียงกันไปแต่ละคน ดังนี้

```
$ mail thisun komsarn suthida@apache01ba
Subject: Meeting Reminder
Dear all,

Please remember that we have a meeting on 13:30 this afternoon.
Don't forget to bring your own report.

And don't forget the pop corn!!!  I guess we would feel hungry
by evening.

pairoj

$
```

สิ่งที่ควรทราบประการหนึ่งเกี่ยวกับโปรแกรม mail ก็คือ ผู้ใช้งานจะไม่มีทางรู้เลยว่ามีจดหมายส่งมาถึงตนหรือไม่ หากไม่ได้เข้าไปตรวจสอบโดยใช้โปรแกรม mail หรือ login เข้าสู่ระบบ UNIX ใหม่ ซึ่งจะได้รับแจ้งเรื่องจดหมายเข้า การที่คอยพิมพ์คำสั่ง mail อยู่บ่อย ๆ เพื่อคอยดูว่ามีจดหมายใหม่เข้ามาหรือไม่ ในขณะที่ต้องทำงานซึ่งติดพันอยู่นับว่าเป็นเรื่องที่ไม่สะดวกนัก ระบบปฏิบัติการ UNIX จึงได้รับการ

ออกแบบให้มีการตรวจสอบจดหมายเข้าใหม่โดยอัตโนมัติทุกช่วงเวลาหนึ่งหลังจากผู้ใช้งานได้ login เข้าสู่ระบบ (โดยทั่วไปมักกำหนดเป็นทุก ๆ 10 นาที) หากตรวจพบว่ามีจดหมายเข้ามาใหม่ ระบบ UNIX จะทำการแจ้งให้ผู้ใช้งานทราบโดยอัตโนมัติด้วยประโยคว่า You have a new mail.

ระบบ UNIX มีคำสั่งชื่อ biff ซึ่งเมื่อผู้ใช้งานป้อนคำสั่ง **biff on** จะทำให้มีการแสดงข้อความบางส่วนของจดหมายที่ได้รับใหม่ เช่น (หัวเรื่องจดหมาย ชื่อผู้ส่ง เป็นต้น) ให้ปรากฏขึ้นบนจอโดยอัตโนมัติ เพื่อให้ผู้ใช้งานตัดสินใจว่าจะหยุดทำงานของตนชั่วคราวเพื่อเปิดจดหมายฉบับนี้อ่านหรือไม่ สำหรับผู้ใช้บางคนที่ไม่ชอบการทำงานของคำสั่งดังกล่าวเนื่องจากการแสดงข้อความบางส่วนของจดหมายจะรบกวนกับงานที่ทำและปรากฏอยู่บนหน้าจอ ก็สามารถสั่งหยุดการทำงานได้โดยการป้อนคำสั่ง **biff off**

แถมท้ายด้วยเกร็ดเล็กน้อย เท่าที่ผู้เขียนทราบมาชื่อคำสั่ง **biff** นี้มิได้ย่อมาจากคำเต็มว่าอะไรหรอกครับ จริง ๆ แล้ว biff เป็นชื่อของสุนัขตัวหนึ่งที่เป็นของนักศึกษาที่มหาวิทยาลัยเบอร์คเล่ย์ซึ่งเป็นสถานที่ที่พัฒนาระบบปฏิบัติการ UNIX จนได้รับการยอมรับมาจนถึงปัจจุบัน เหตุที่มีการนำเอาชื่อของเจ้าสุนัขตัวนี้มาเป็นชื่อชุดคำสั่งเกี่ยวกับการรับอิเล็กทรอนิกส์เมล์ก็เนื่องมาจากกิตติศัพท์ของมันที่คอยไล่งับบุรุษไปรษณีย์ทุกคนที่มาส่งจดหมายนั่นเอง

ฉบับหน้าผู้เขียนจะได้กล่าวถึงการจัดโครงสร้างไฟล์ของระบบปฏิบัติการ UNIX พร้อมกับชุดคำสั่งที่เกี่ยวข้องกับการจัดการไฟล์ต่อไป พบกันเดือนหน้าครับ

เทคนิคการใช้ Drag & Drop บนวินโดวส์ 95, NT

การใช้งานโปรแกรมบนวินโดวส์ 95, NT ส่วนใหญ่เราจำเป็นต้องพึ่งพาการใช้เม้าส์ ซึ่งการทำงานที่เกี่ยวข้องกับการใช้เม้าส์นั้น ก็จะหนีไม่พ้นการ Drag & Drop หรือวิธีลากแล้วปล่อย ซึ่งมีเกร็ดเล็ก ๆ น้อย ๆ มาฝาก ดังนี้

1. ในระหว่างที่ลาก ถ้าหากเม้าส์พอยน์เตอร์มีเครื่องหมายบวก (+) แสดงว่าเป็นการคัดลอก แต่ถ้าเป็นลูกศร (↗) แสดงว่าเป็นการคัดลอกคีย์ลัด (Shortcut)

2. ในระหว่างที่ลาก ถ้าหากกดปุ่ม ESC จะเป็นการยกเลิกทันที

3. ถ้าต้องการสร้างคีย์ลัด (Shortcut) เพื่อเรียกใช้งานโปรแกรม คุณสามารถคลิกไปยังไฟล์โปรแกรมที่ต้องการ แล้วลากไปไว้ในตำแหน่งที่คุณสะดวกต่อการเรียกใช้

4. ถ้าลากไอคอนที่อยู่บนเดสค์ท็อปไปวางที่เมนูเริ่ม (Start Menu) ก็จะพบว่ามีไอคอนของโปรแกรมดังกล่าวอยู่ด้านบนสุดของเมนูเริ่ม เพื่อให้เรียกใช้งานได้โดยทันที

5. การลากโดยคลิกเม้าส์ซ้ายจะไม่มีการถามยืนยันว่าคุณต้องการทำอะไร การทำงานขึ้นอยู่กับคุณว่าจะทำงานร่วมกับคีย์อะไร เช่น Ctrl หรือ Shift หรือการลากเอาไปไว้ในตำแหน่ง แต่ถ้าหากเป็นการลากโดยคลิกเม้าส์ขวา โปรแกรมจะมีไดอะล็อกซ์แสดงตอนช่วงที่คุณวางว่าต้องการ ทำการคัดลอก ย้าย หรือสร้างคีย์ลัด (Shortcut) โดยคุณสามารถเลือกที่จะยกเลิกก็ได้

6. การกดปุ่ม Shift และคลิกเม้าส์ซ้ายเพื่อเลือกกลุ่มไฟล์ทีละหลาย ๆ รายการ คุณก็สามารถเปลี่ยนเป็นคลิกเม้าส์ขวาโดยเม้าส์พอยน์เตอร์จะต้องไม่ขยับออกไปนอกขอบที่เลือก จากนั้นลากไปปล่อยยังตำแหน่งที่ต้องการ โดยคุณจะได้รับความสะดวกในการเลือกใช้เมนูลัดที่ได้จากการคลิกขวา